Session 4: Simulation Environments

Multiprocessor Architecture Evaluation using Commercial Applications
Ashwini K. Nanda
IBM TJ Watson Research Center

CASE - A Computer System Simulation Environment for Commercial Workloads
Jim Nilsson, Fredrik Dahlgren, Magnus Karlsson, Peter Magnusson* and Per Stenstrom
Department of Computer Engineering, Chalmers University of Technology
*Swedish Institute of Computer Science

VPC and IADYN - Project Overviews
Rich Uhlig and Todd Austin
Intel Corporation

Multiprocessor Architecture Evaluation using Commercial Applications

Ashwini K. Nanda

IBM T.J. Watson Research Center
P.O. Box 218 Yorktown Heights, NY 10598
ashwini@watson.ibm.com

Project: *Commercial Server Architecture and Performance*

Contributors:

Ashwini Nanda

Moriyoshi Ohara

Yiming Hu

Maged Michael

Caroline Benveniste

Motivation

1. Shared Memory Multiprocessors: dominant architecture for future enterprise market
    - Message Passing not acceptable to programmers.
    - Small and medium SMPs (shared memory) determine volume

2. Commercial Applications will drive the architecture of scalable shared memory systems
    - Database: Decision Support, Transaction Processing
    - Web Servers

3. Must understand memory reference behavior and its impact on performance
    - Most studies have focused on technical applications: SPLASH2
        Regular, Predictable, smaller memory footprints
    - Commercial applications: complex, many have large memory footprints

Difficulties in Evaluating Commercial Applications

1. Commercial applications spend ***significant*** amount of time in the OS
2. Most execution driven simulators ***do not model OS activity***

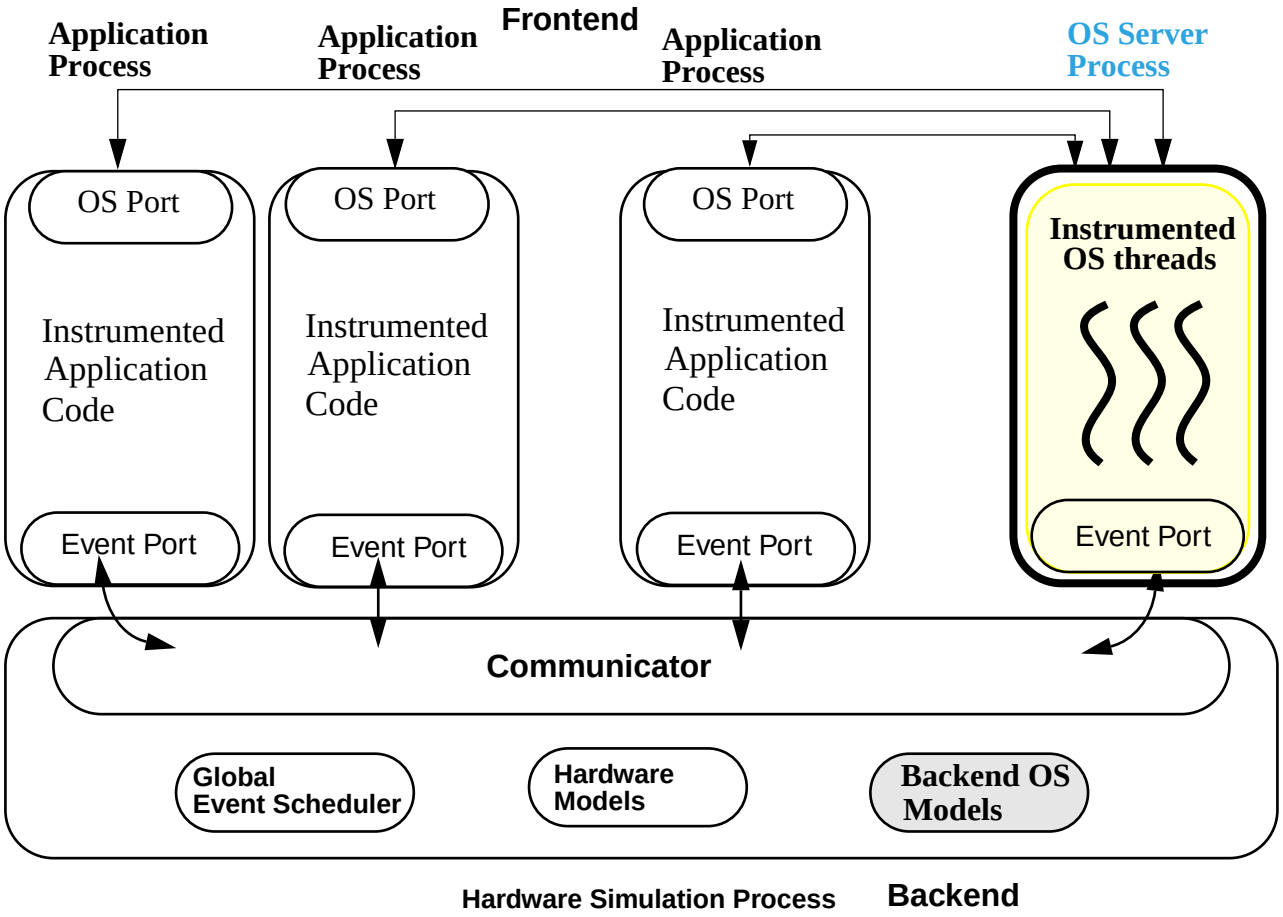
benchmark	**user time**	**OS time**		
		total	interrupt handlers	kernel
SPECWeb/Apache	14.9%	**85.1%**	37.8%	**47.3%**
TPCD/DB2 (100MB data base)	81%	**19%**	8.6%	**10.4%**
TPCC/DB2 (400MB data base)	79%	**21%**	14.6%	**6.4%**

3. Popular commercial application code is ***typically proprietary***
4. Applications such as Web servers are ***relatively new***

COMPASS

COMmercial **PA**rallel **S**hared Memory **S**imulator:

Execution driven simulator that models ***significant OS activity***



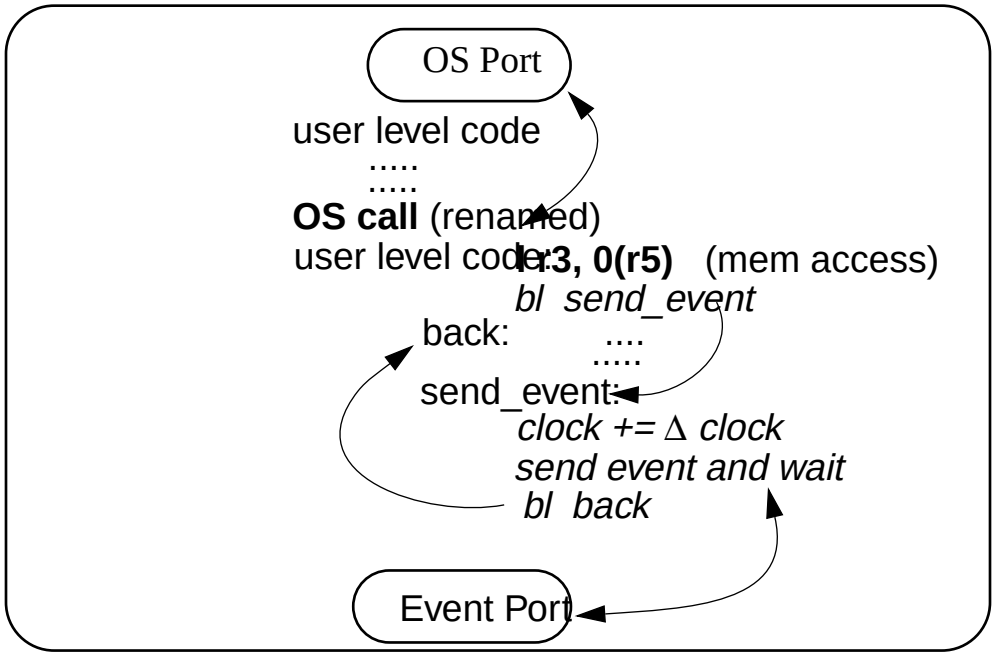
The OS Server

Select Important OS functions through profiling: *category 1 functions*

For TPC-C, TPC-D on DB2 and SPECWeb on Apache:

kwritev, kreadv, select, statx, connect, naccept, send, mmap, munmap

Modify corresponding OS code to run in user mode



OS functions and Devices in the *backend*

Category 2 functions : Indirectly affecting memory access behavior

File I/O : mmap, munmap

Virtual memory management: simulate shmget, shmat and shmdt

page translation, page placement & page migration policies

Process Scheduling: simulated mapping of processors to processes

FCFS, preemptive, affinity scheduling

Physical Devices:

clock interrupt, ethernet, hard disk drive and controller

Comparison of Simulation Environment

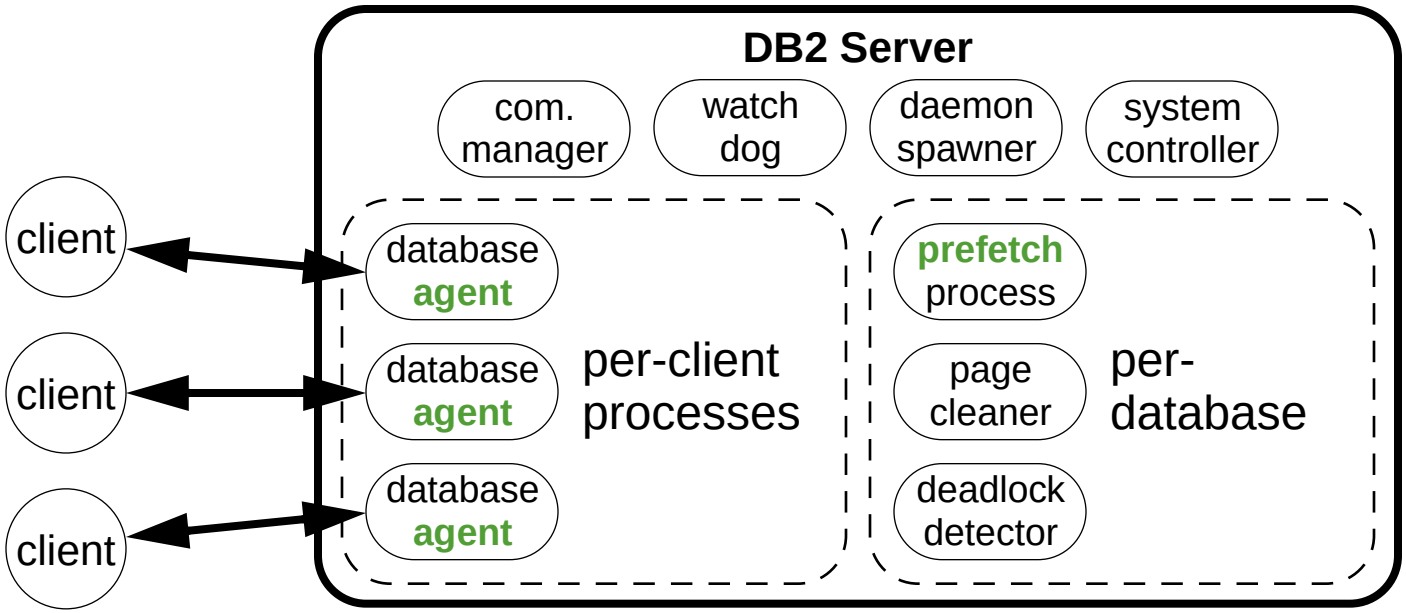
	Augmint	Mint	Tangolite	SimOS	COMPASS
Simulation Technique	Code Augmentation	Native execution and interpretation	Code Augmentation	Native execution, Binary Translation	Code Augmentation
Programming Model	thread	process	thread	process	process
OS/Device Simulation	none	none	none	yes	yes
OS Simulation Accuracy	N/A	N/A	N/A	very high	high
Porting a new OS	N/A	N/A	N/A	relatively difficult	relatively easier
Porting a new commercial application	difficult	difficult	difficult	easy	requires minor changes
Simulation slowdown factor	not available	30-60 (empty h/w model)	not available	5-27000	300-600

Example application: DB2

DB2 Common Server Version 2

- A Relational Database Management System
- SMP support (shared-everything model) for Inter-Query Parallelism

Server Process Structure **(not all processes are shown)**



- Instrumented all server processes
- Database agent: the main query processor

TPC-D Queries

Benchmark for *Decision Support Applications*

- Complex queries
- Mostly read-only queries

Sample query (Q3)

- retrieves the unshipped orders of customers within a market segment and dates
- several primitives (sort, join, sequential scan, index scan)

```
SELECT
    L_ORDERKEY, SUM(L_EXTENDEDPRICE * (1 - L_DISCOUNT)) AS REVENUE
    O_ORDERDATE, O_SHIPPRIORITY
    FROM CUSTOMER, ORDER, LINEITEM
    WHERE  C_MKTSEGMENT = '[segment]'
        AND C_CUSTKEY = O_CUSTKEY
        AND L_ORDERKEY = O_ORDERKEY
        AND O_ORDERDATE < DATE '[date]'
        AND L_SHIPDATE > DATE '[date]'
    GROUP BY L_ORDERKEY, O_ORDERDATE, O_SHIPPRIORITY
    ORDER BY REVENUE DESC, O_ORDERDATE;[1]
```

1. Copyright Transaction Processing Performance Council 1993, 1995, TPC Benchmark D, 19 December 1995, copied by permission of the Transaction Processing Performance Council

Example System Configuration

Database : DB2 Common Server Version 2, DB size : 12MB-400MB,

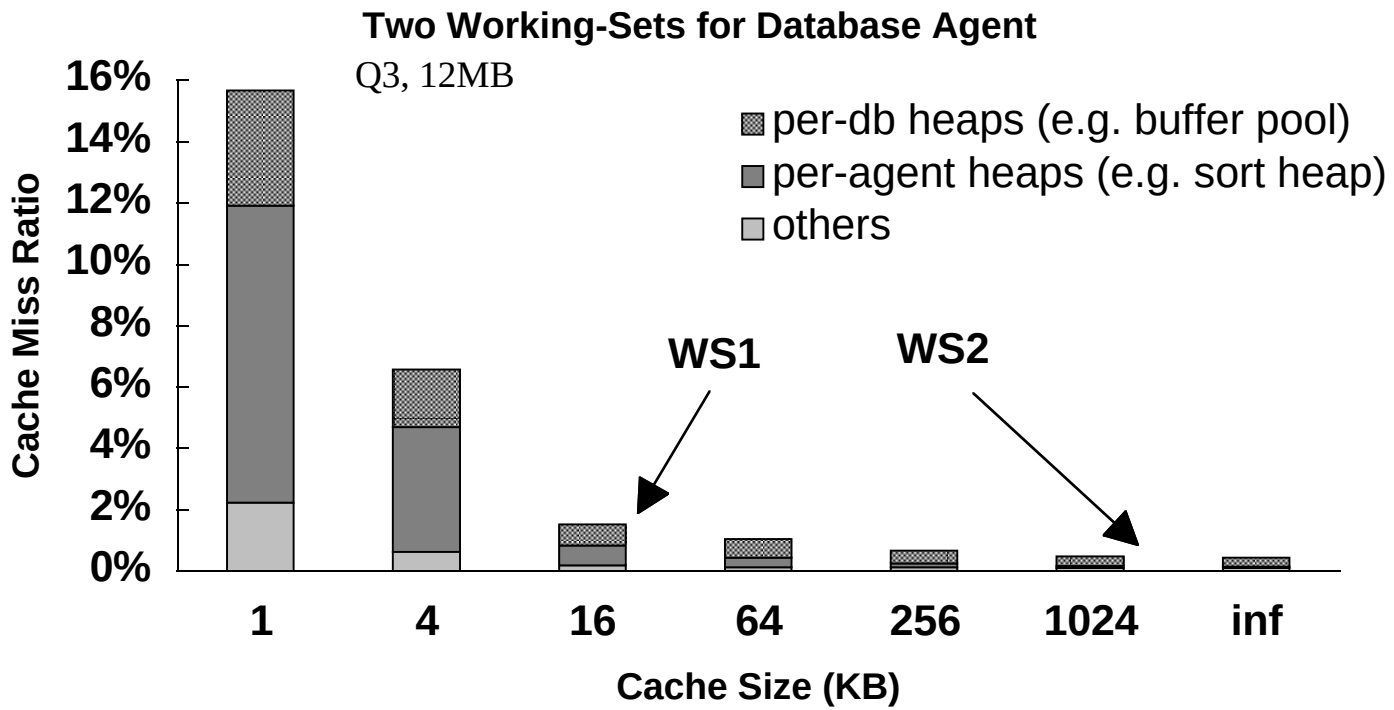
Buffer Pool = 1/10 of the Database Size, TPC-D Q3,

System : Dedicated processor/process, PRAM memory model for faster

Page Placement : Round Robin on shared data

Cache : 64B Line, direct mapped to full-associative LRU, cache size 1K-inf

Working-Set: Base Characteristics

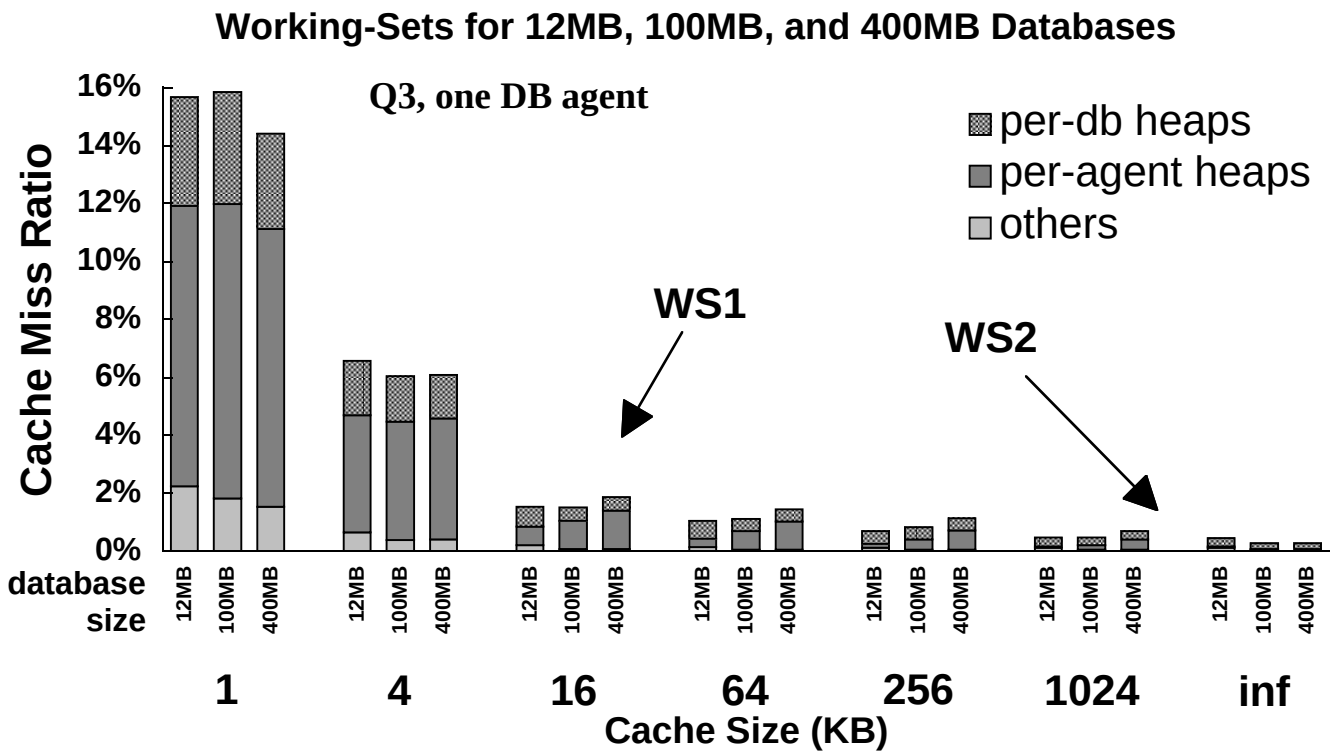


Working set 1 (WS1) : Majority of per-agent heaps, around 16k-64K

Working set 2 (WS2) : The complete data set, > 1MB

Low overall miss rates with comparatively small caches

Working-Set: Effect of Database Size

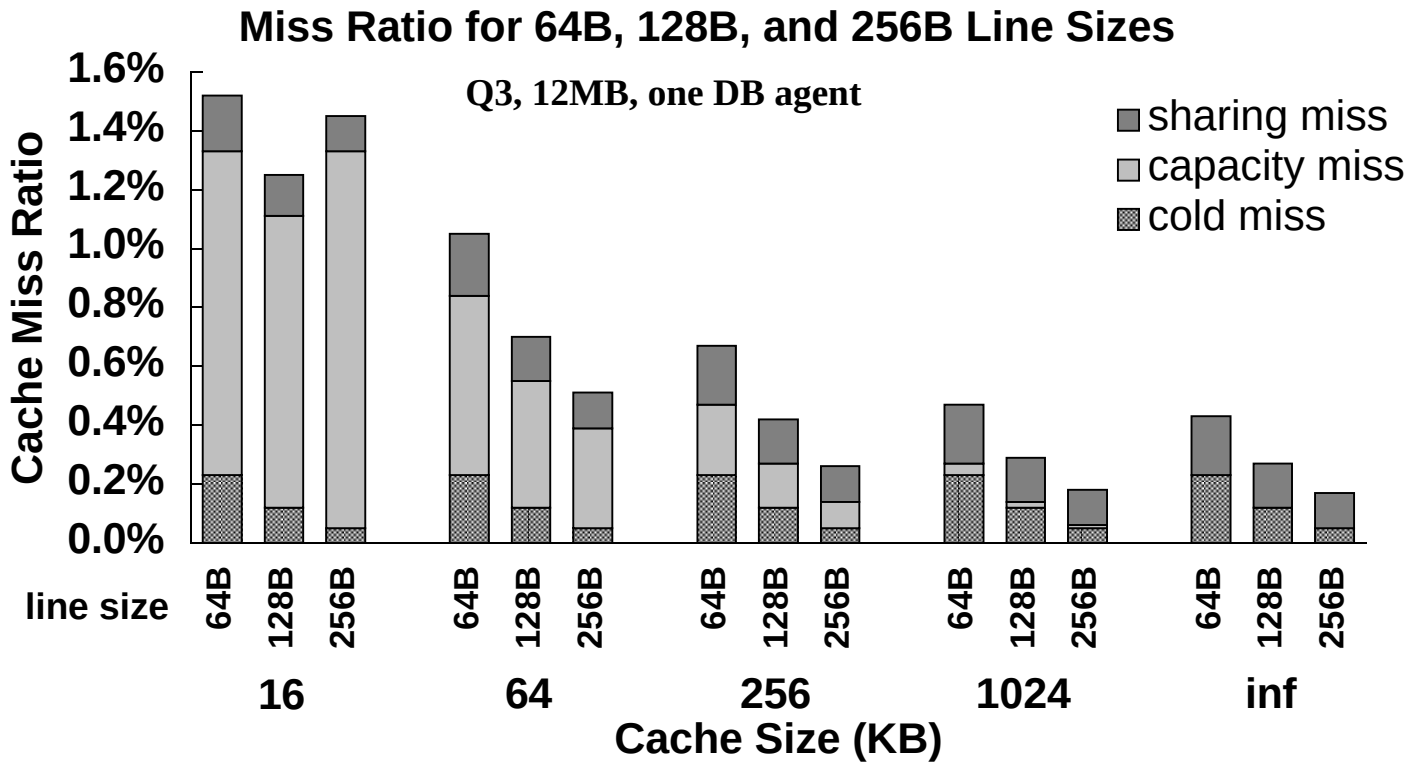


Miss ratios remain almost constant with increasing DB size

WS1: relatively independent of DB size

WS2: increases with DB size, but very low overall cache miss ratio

Effect of Large Line Size



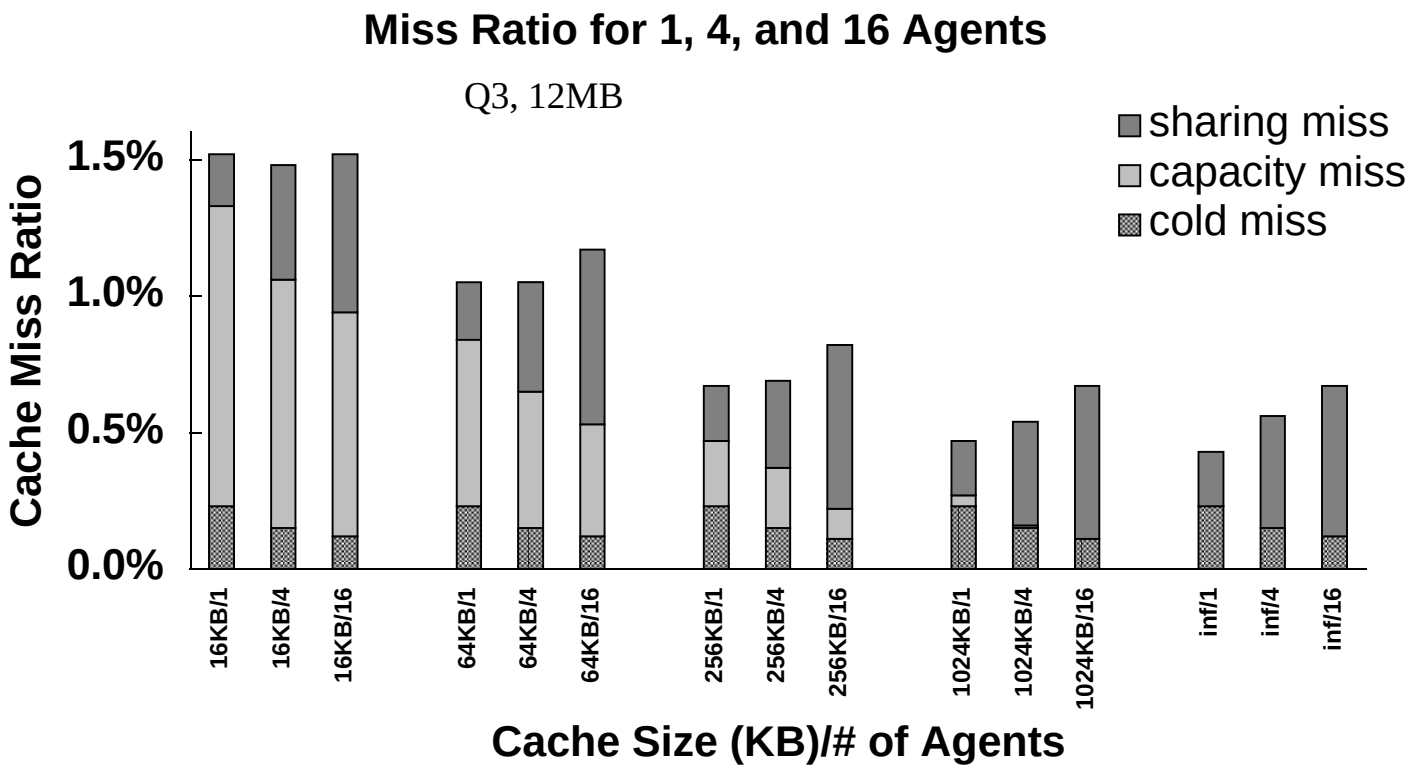
Poor spatial locality for shared data structures

Large lines help cold misses

knee at 128B for smaller caches

Larger lines up to 256B continue to yield better hit rates

Communication Characteristics

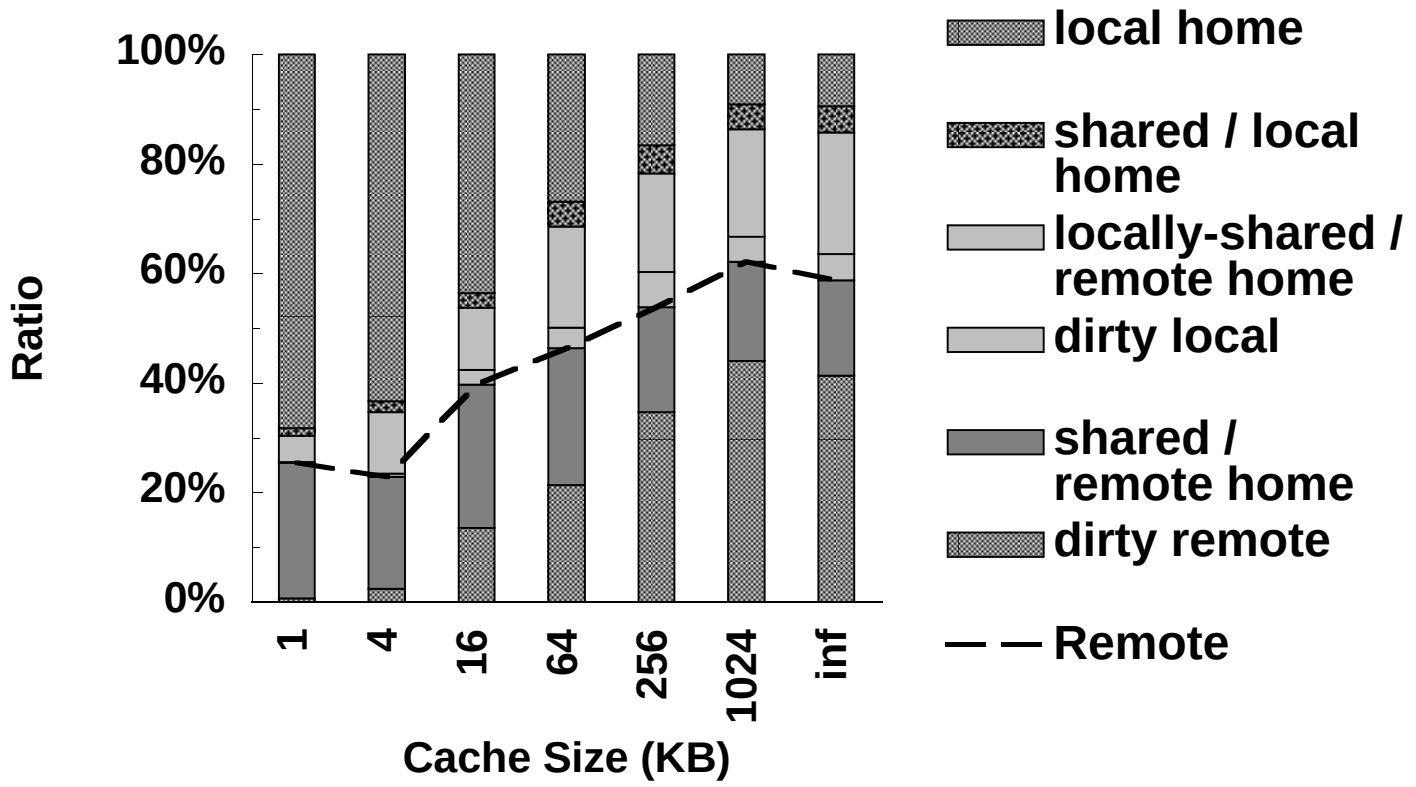


Sharing misses increase with # of DB agents

Overall cache misses increase with # of DB agents

Local vs. Remote accesses in a NUMA system

Configuration : 12MB TPC-D Q3, 16 Database-Agents, 4 processors per node



40% or more of the cache misses are satisfied locally due to large amount of read sharing **(with uniform local/remote page distribution)**

Summary

COMPASS: Commercial Parallel Shared Memory Simulator

OS functions modeling

Execution driven simulation results for TPC-D/DB2

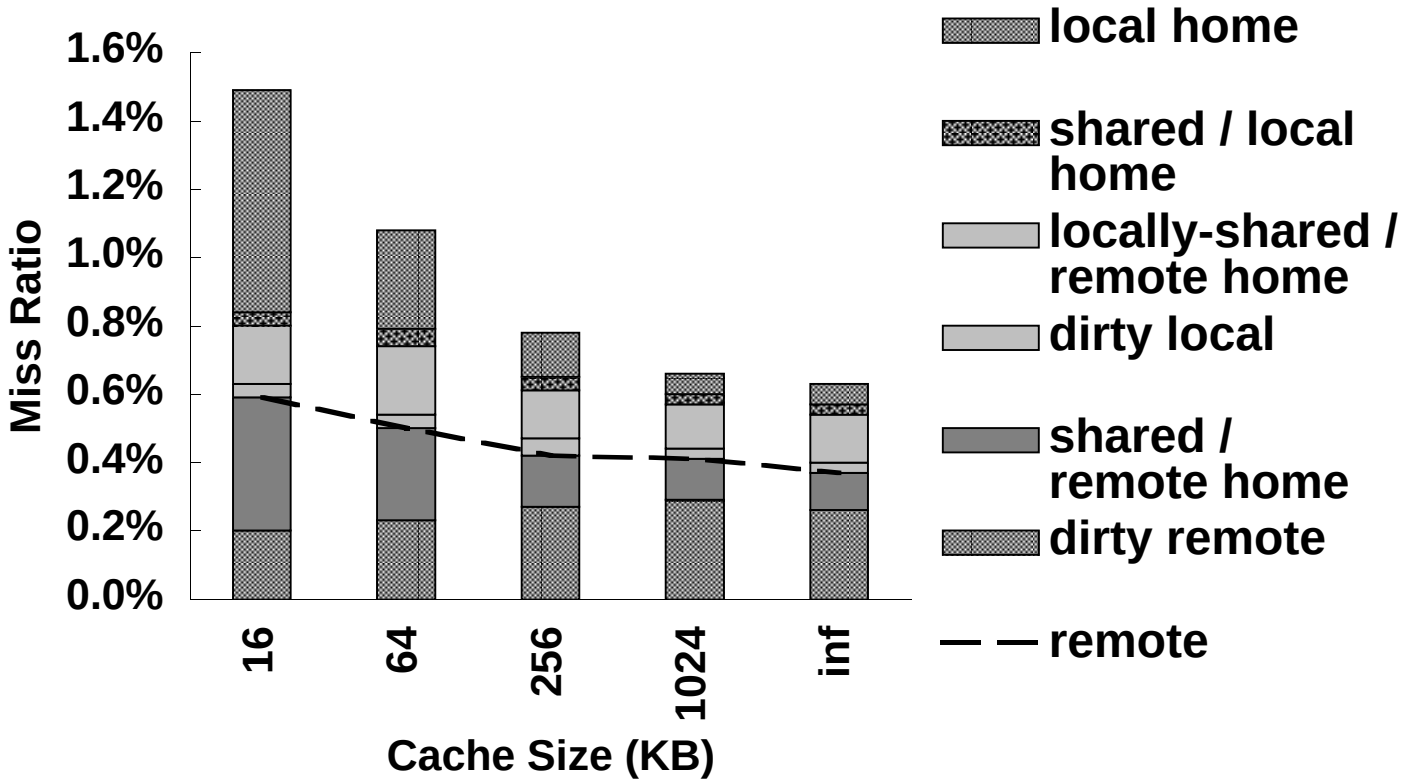
Two distinct working sets for TPC-D

Significant working set is very small (~ 16KB)

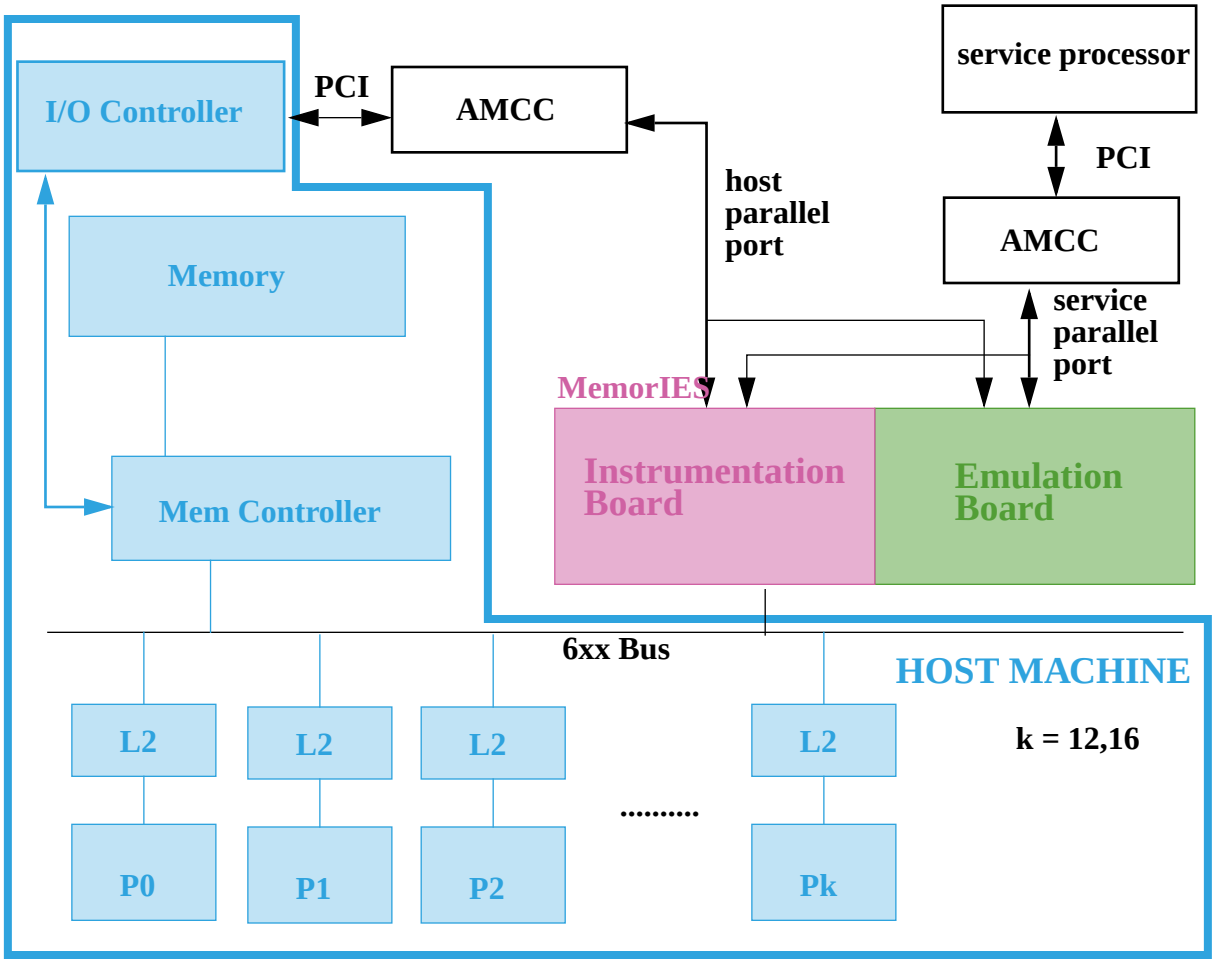
Miss ratios and significant working set are nearly independent of DB size

Overall miss ratios increase marginally with number of processes (processors)

Good local/remote ratio due to large read sharing


1.6%
1.4%
1.2%
1.0%
0.8%
0.6%
0.4%
0.2%
0.0%
Miss Ratio
16
64
256
1024
inf
Cache Size (KB)
local home
shared / local home
locally-shared / remote home
dirty local
shared / remote home
dirty remote
remote

MemorIES : Memory Instrumentation and Emulation System



Current and Future Work

Execution time studies with detailed hardware models

Investigating *intra-query* parallelism in DB2

Investigate behavior of TPC-C

Web Servers: Apache

Multimedia and data mining applications

Computer System Evaluations with Commercial Workloads based on SimICS

Fredrik Dahlgren, Jim Nilsson, Magnus Karlsson,
Fredrik Lundholm, Per Stenström
Chalmers University of Technology

Peter Magnusson, Fredrik Larsson, Andreas Moestedt, Bengt Werner
Swedish Institute of Computer Science

Håkan Grahn
University of Karlskrona/Ronneby

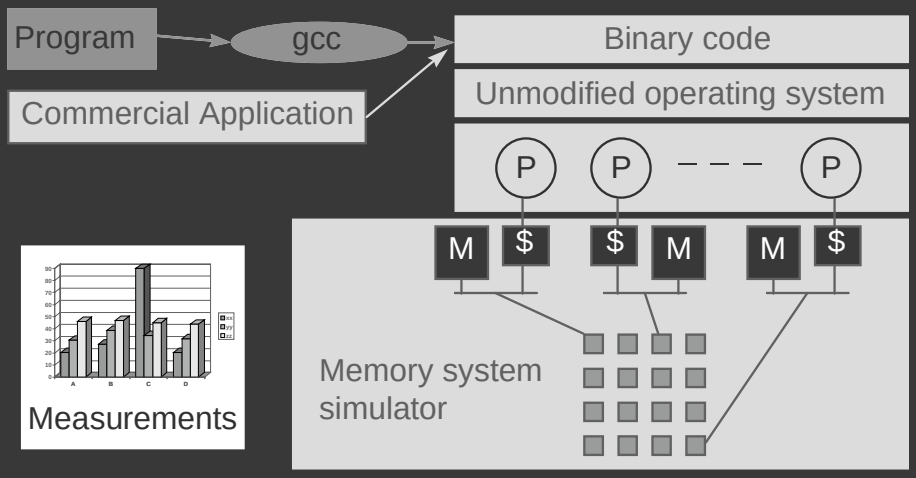
Simulator Requirements

Database vs. Scientific:
- Larger data sizes
- More interaction with the operating system
- A substantial amount of disk activity
- The memory management and buffering important

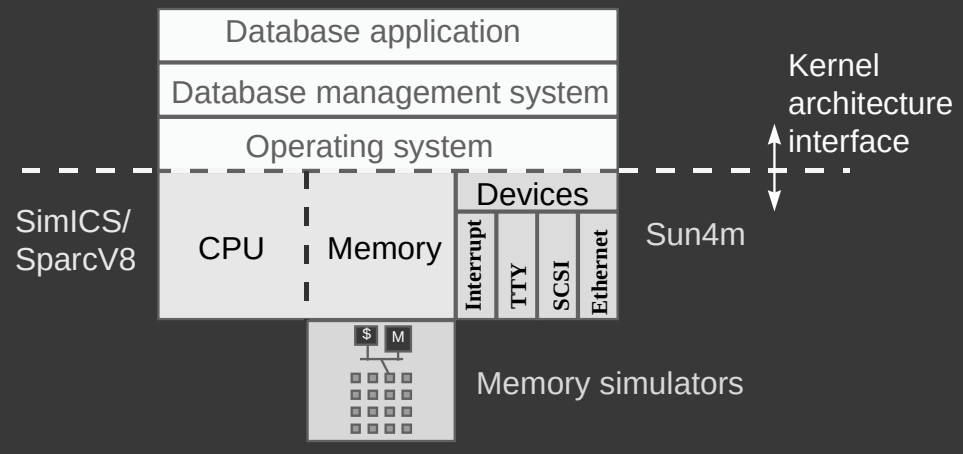
Inclusion of operating system imperative!

Usage:
- Architecture evaluations of given application
- Performance tuning of application on given architecture
- Evaluating modification of HW/SW interface and its hardware support as well as software consequences

Goals for our Simulation Platform



SimICS/Sun4m Overview



Currently executes unmodified Linux 2.0.30 and Solaris 2.6!

SimICS

Sparc V8 instruction-set simulator
Supports:
- Multiple processors
- Multiple address spaces
- System-level code

Profiling and symbolic debugging of user/system code

Slowdown (SPECint95): 25-80 depending on statistics

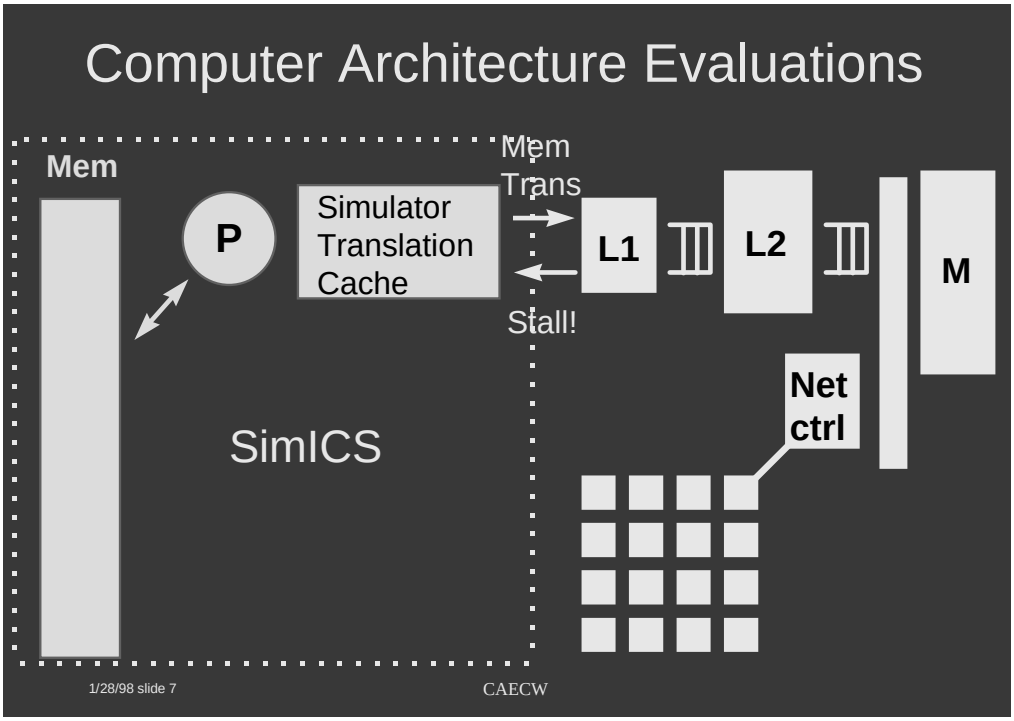
Allows the writing of separate modules to simulate:
- devices,
- memory management units,
- memory systems

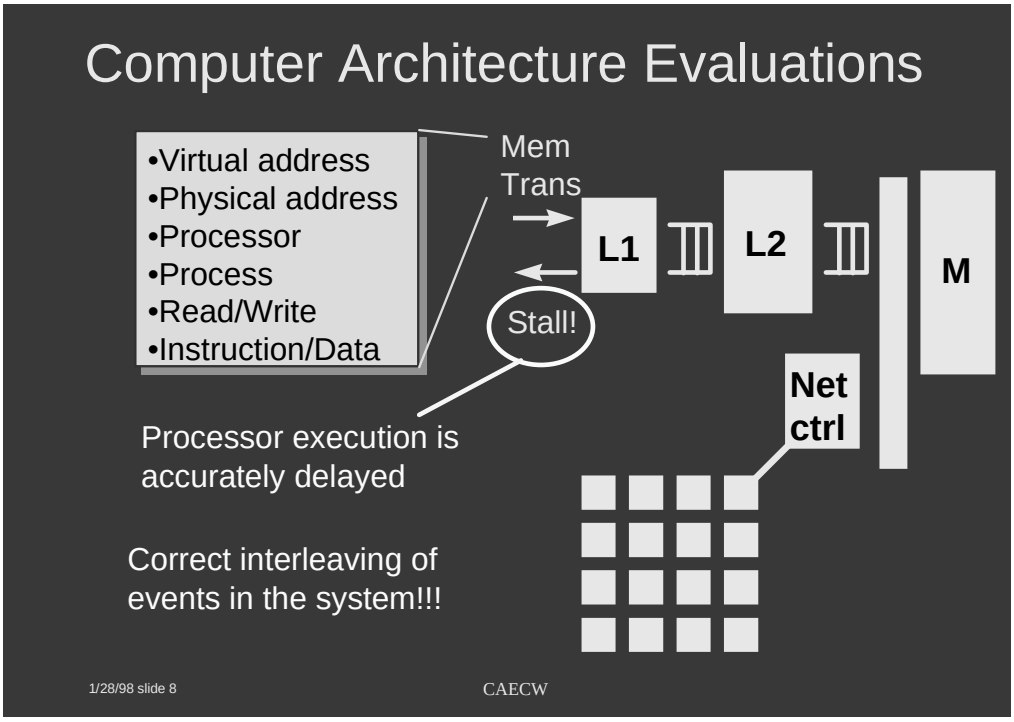
Modeling the Kernel Architecture

All device simulators have been developed so that Linux 2.0.30 as well as Solaris 2.6 boot and run completely unmodified.

PROM:
- Reverse-engineered
- Short-circuits the boot-PROM phase
- Sets up device and architecture structures by parsing a target architecture description

SCSI:
- Complex: Highly asynchronous
- Several different tasks can be simultaneously pending
- Disk contents: Dumps of real partitions


Computer Architecture Evaluations
Mem
P
Simulator
Translation
Cache
Mem
Trans
Stall!
L1
L2
M
Net
ctrl
SimICS
1/28/98 slide 7
CAECW


Computer Architecture Evaluations
•Virtual address
•Physical address
•Processor
•Process
•Read/Write
•Instruction/Data
Mem
Trans
L1
L2
M
Stall!
Net
ctrl
Processor execution is
accurately delayed
Correct interleaving of
events in the system!!!
1/28/98 slide 8
CAECW

Example: Evaluating Memory Systems using TPC-D

Linux 2.0.30, PostgreSQL from UC Berkeley, TPC-D 1/50 Q6 Simulations on a Sun Enterprise Server 4000

	EXE.TIME	*SLOWDOWN*
native, SparcStation4	0:16 min	
simple 1-proc simulation	16:33 min	62
simple 4-proc simulation (same query on each)	1:15 h.	281 (70)
detailed 4-proc. simulation	2.11 h.	491 (123)

Example: 8-processor Sun4m

Sun4m is only specified for up to 4 processors.

Task: Modify Linux 2.0.30 for Sun4m as well as the architecture description for 8 processors.

Architecture:

- Create 4 new instances of some devices,
  e.g. interrupt and counter, by modifying PROM tree.

Linux:

- Interrupt handling, processor identification mechanisms
- Trap-base register
- Simulation: symbolic debugging of operating system

Open Issues

- How relevant are down-scaled database experiments?
  - Databases are scaled down 2-4 magnitudes
  - How is the memory access behavior affected?
  - Can realistic memory buffering or paging effects be included?
- How useful are public database handlers (PostgreSQL)?
  - Comparisons against commercial database handlers

Summary

SimICS/Sun4m currently executes unmodified Linux 2.0.30 and Solaris 2.6.

Symbolic debugging of application as well as kernel code.

The slowdown simulating database applications is 62 ->.

Simple interface for memory system simulators for uniprocessor as well as multiprocessor simulations.

Possibility for evaluations of system architecture, such as memory system organization, applications code, HW/SW interface (e.g. prefetching, bulk data transfer, ...)

Performance Analysis Tools for PC-based Systems

January 1998

Intel Microcomputer Research Lab

Rich Uhlig

Todd Austin

Introduction

- Intel Microcomputer Research Lab
  - Charter: Identify and remove future PC-system bottlenecks
- Our group's focus
  - To develop tools and infrastructure for analyzing iA-based systems
- Current State of the Art
  - Much excellent work in analysis tools for RISC-based systems
  - Few system-analysis tools available for iA-based systems
- This Talk: Overviews of works-in-progress...
  - The VPC Project: A PC platform simulator
  - The iADyn Project: Flexible iA32 ISA simulators

Why is analyzing PC architectures hard?

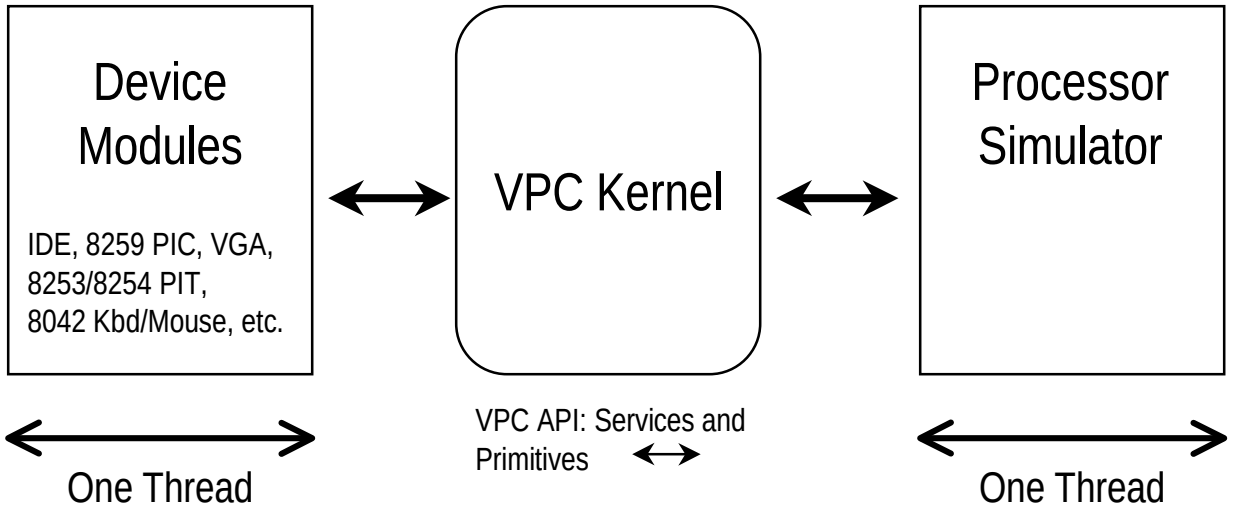
- Instruction Set Architecture: A Moving Target
  - Every generation of x86 implementation has had ISA extensions
- Extreme Diversity in Platform Components
  - System Hardware Architectures
    - NT distinguishes > ~4,500 iA32-based systems (compare with 200 RISC-based systems)
  - Bus and Firmware Standards
    - ISA, EISA, PCI, SCSI, AGP, USB, 1394, ACPI, PnP, etc.
  - Devices
    - More than 200 display adapters, 750 storage devices, 300 network adapters
    - NT recognizes more than 4,000 devices
- Many PC Operating Systems
  - WinNT, Win95, Win98, OS/2, UnixWare, Linux, Solaris, Mach, etc.

Our Goals

- We want to understand interplay between:
  - Future ISA extensions and processor micro-architecture
  - Future platform architecture
  - New applications and system software
- Problem:
  - No single group can hope to model such broad functionality
- Our Strategy
  - Build extensible tool sets
  - Put flexibility before raw speed
  - Build generic models and tools, document them, and set them free...

VPC: A PC Platform Simulator

- Based on a small, simulator kernel
  - Provides generic abstractions for modeling device interrupts, DMA, I/O ports, memory-mapped I/O, etc.
- First-generation prototype
  - Models basic AT-compatibility devices
  - Boots full set of standard NT binaries in about 5 minutes

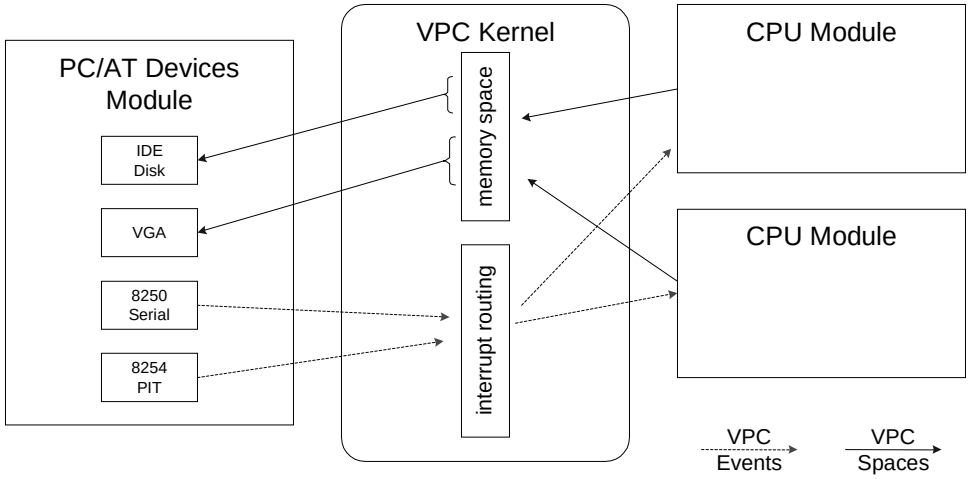


VPC Applications

- 100% Software-based system tracing tool
  - Hardware-based tracing is costly and fragile
  - Software-based is cheaper, more reliable, and more detailed
- On-the-fly driver for trace-driven analysis tools
  - Cache simulators
  - Processor u-arch simulators
- Coarse-grained System Software Performance Analysis
  - Quick turn-around on “what-if” scenarios
    - What if {2x, 5x, 10x} {larger, smaller, faster, slower} {memory, disk, processor}
    - Running actual NT binaries => realistic responses from system software

The VPC Abstractions

- The VPC kernel is a “switchbox” between platform devices
- Supports three basic abstractions
  - Spaces: for modeling I/O port space, physical memory space, disk images
  - Events: for modeling interrupts, performance events
  - Time: for synchronizing interactions between devices in the temporal domain

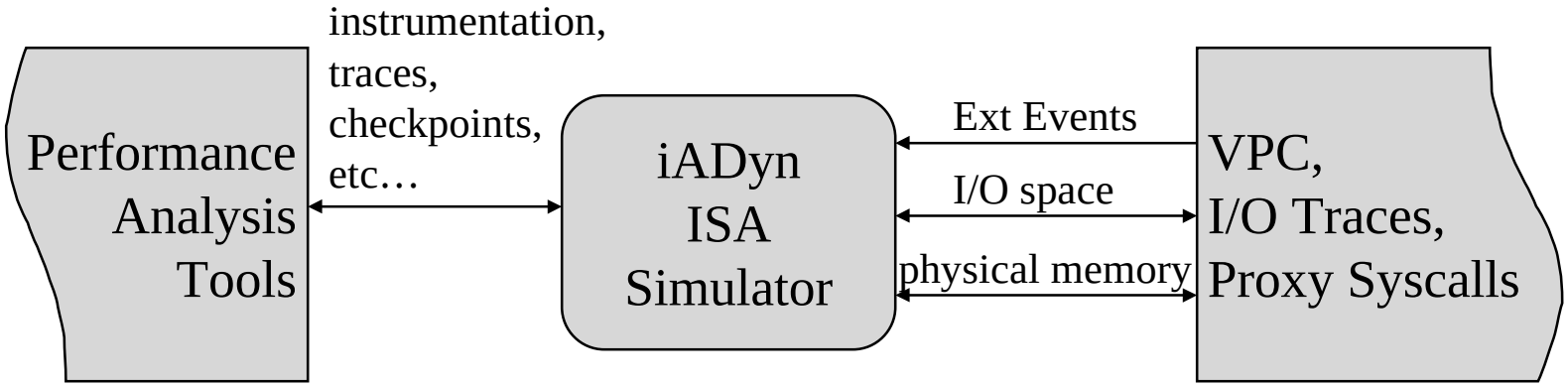


VPC Future Directions

- Add breadth and depth to device models
  - ISA, EISA, PCI, SCSI, AGP, USB, 1394, ACPI, PnP, etc.
  - Working with design teams inside company to plug models into VPC kernel
- Tighter integration with performance simulators
  - Close feedback between cycle-accurate performance simulators
  - Retain ability to run in “fast functional” mode
- Processor and chipset design validation
  - Use VPC as a checker running in parallel with RTL models
  - Use sampling to cover large span of workload activity
- Pre-silicon system software development
  - BIOS development for new platforms
  - Device driver debugging for new devices

iADyn: iA32 ISA Simulators

- Developing a family of iA32 functional simulators
  - implement iA32 instruction execution with varying degrees of precision
- Designed to serve wide variety of users
  - workload positioning - light tracing - detailed microarchitecture simulation
- Implementation based on a formal model of iA32 ISA
  - DEFIANT development framework encodes iA32 syntax and semantics
- Initial prototype serves detailed microarchitecture simulation



iADyn Applications

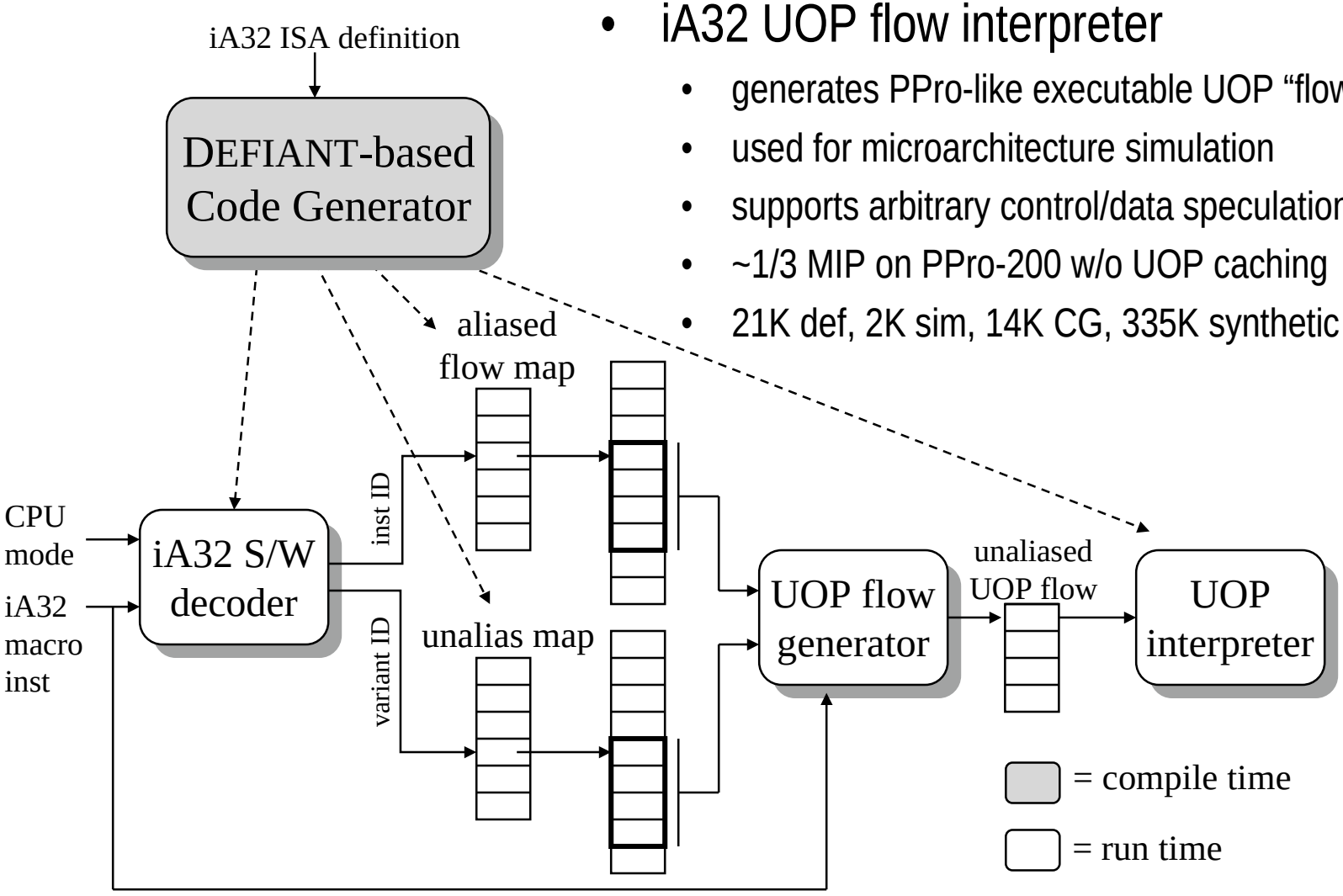
- Fast functional simulation
  - workload positioning, system Monitoring, light tracing
  - requires fast execution
- Instruction set design
  - prototype experimental instructions on S/W CPU component
  - requires flexible, extensible ISA simulation framework
- Detailed microarchitecture simulation
  - used to drive microarchitecture model simulators
  - requires decomposition of macro-instructions into micro-instructions (UOPs)
  - requires support for arbitrary control/data speculation at UOP boundaries
- Observation: no single implementation serves all these applications well

Managing Complexity with DEFIANT

- Quantifying the complexity:
  - different application require different simulator implementations
  - over 700 instructions with more than 15000 semantically unique variants
- Yet, all these simulators all implement the same ISA
- Solution: The DEFIANT development framework
  - rapid code development framework for iA32-specific code
  - based on formal definition of iA32 ISA syntax and semantics
  - meta-programming interface used to probe iA32 ISA model data
  - iA32 formal ISA definition reused across all developments

Initial Prototype: Functional Flow Interpreter

- iA32 UOP flow interpreter
  - generates PPro-like executable UOP “flows”
  - used for microarchitecture simulation
  - supports arbitrary control/data speculation
  - ~1/3 MIP on PPro-200 w/o UOP caching
  - 21K def, 2K sim, 14K CG, 335K synthetic



Summary

- Lack of tools for PC-based system performance analysis
  - complex, evolving instruction set
  - diverse platform terrain
- Two works-in-progress aim to fill this void:
  - The VPC Project: A PC platform simulator
  - The iADyn Project: Flexible iA32 ISA simulators
- Tough problem to solve alone, benefits from:
  - extensible software designs that facilitate collaboration
  - flexible software architectures that enable broad application of tools